情報システムの取引慣行・契約に関する実施ガイド
―パッケージやサービス活用モデル（企画・開発）―
（ドラフト）

前回からの議論
- SaaS について（ヒアリング）
  チェックリストを導入するのは各社とも賛成している
  不良ベンダの排除になると考えられる
  セキュリティに関しては ISMS とか言うのではなく、軽いものがほしい
  ベンチャーの参入を阻害してはいけない

本資料の変更状況
・アジャイル部分の差し替え
・モデルプロセス変更に伴う変更中で、中途半端な状況になっています

平成 19 年 9 月 25 日

社団法人コンピュータソフトウェア協会（CSAJ）
社団法人日本コンピュータシステム販売店協会（JCSSA）
事務局

目　次

１．　総論 1
１．１ 経緯 1
１．２ 目的 1
１．３ ガイドの全体像 1
１．４ 用語 4
２．　パッケージベース開発 5
２．１ 概要 5
２．２ モデル契約プロセス 7
２．２．１ 企画プロセス 7
２．２．２ 要件定義 9
２．２．３ 開発 9
２．３ 企画・要件定義・開発で起こるトラブル例と必要な対応 11
２．４ 関係者の役割分担 18
２．５ ベンダの説明事項と手順 20
２．５．１ 企画・要件定義プロセス 20
２．５．２ 開発プロセス 30
２．６ パッケージインテグレーション型での補足事項 34
２．７ ミドルウェア・インフラパッケージの利用での補足事項 34
２．８ 必要なドキュメント 34
３．　サービス調達（ＡＳＰ、ＳａａＳ） 35
３．１ 概要 35
３．２ 主要プロセス 36
３．３ 企画・開発で想定されるトラブル等 39
３．４ 関係者の役割分担 41
３．５ ベンダの説明責任事項 41
３．６ 必要なドキュメント 41
４．　開発手法別の留意点 42
４．１ アジャイル開発・プロトタイピング 42
４．１．１ 概要 42
４．１．２ 主要プロセス 42
４．１．３ 企画・開発時の要点 43
４．１．４ 関係者の役割分担 44
４．１．５ ベンダの説明責任事項 46
４．２ 繰り返し型開発 48
４．２．１ 概要 48
４．２．２ 主要プロセス 48
４．２．３ 企画・開発で起こるトラブル 48
４．２．４ 関係者の役割分担 49
４．２．５ ベンダの説明責任事項 49
５．　モデル契約書・逐条解説 50

付録

付録１　コンサルティング会社選定のためのチェックリスト
付録２　提案依頼書のチェックリスト
付録３　ユーザＩＴ成熟度評価チェックシート
付録４　業務システム仕様書の記述レベル
付録５　パッケージ選定のためのチェックリスト
付録６　SaaS 選定のためのチェックリスト
付録７　検収チェックリスト

委員一覧

平本	健二	フューチャーアーキテクト株式会社 経営企画室　シニアディレクター
浅野	正樹	株式会社 ＣＳＫホールディングス 法務部　知的財産管理課　課長
水谷	学	ピー・シー・エー株式会社 取締役副社長　（公認会計士）
小西	理	株式会社　豆蔵　IT 戦略支援事業部 事業部長
小林	陽二郎	独立行政法人情報処理推進機構 エンタプライズ系プロジェクト（開発プロセス共有化）研究員
前川	徹	サイバー大学　　IT 総合学部　教授
佐藤	裕子	ピー・シー・エー株式会社 総務部法務担当
伊藤	孝洋	株式会社オービックビジネスコンサルタント 管理部
川手	真史	株式会社内田洋行 情報システム事業部企画部システム企画課課長
倉石	英一	IT マネジメント・サポート協同組合（ＥＲＰ研究推進フォーラム） ＩＴコーディネータ協会（紹介）
瀬戸口	靜美	マイクロソフト株式会社　業務執行役員 テクニカルソリューション推進統括本部長

１．総論

１．１ 経緯

2007 年 4 月 13 日に経済産業省から「情報システムの信頼性向上のための取引慣行・契約に関する研究会」最終報告書”情報システム・モデル取引・契約書”が発表された。その中で、パッケージを中心としたシステム導入の場合や反復繰り返し型の開発の場合、中小企業ユーザにおける活用の場合等、本モデル取引・契約書で十分カバーできていない論点について、業界団体を中心としてさらに議論が深められることが望まれると指摘されている。よって、2006 年に実施してきた CSAJ/JCSSA 共同作業部会を発展させ、情報システム・モデル取引・契約書に関する検討を実施する委員会を設置し、検討を進めることとなった。

１．２ 目的

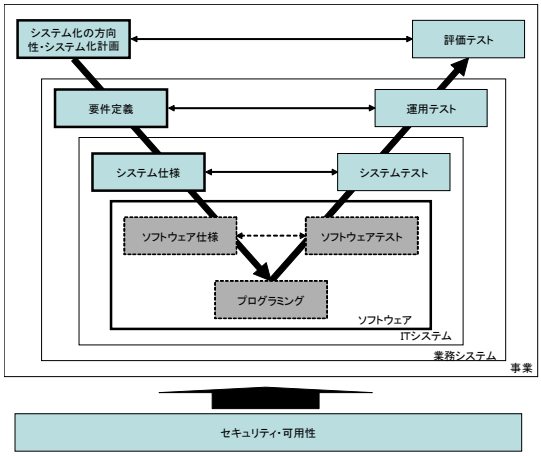
「情報システムの信頼性向上のための取引慣行・契約に関する研究会」最終報告書を踏まえ、パッケージソフトウェア、ASP を利用し情報システムを構築する中小・中堅企業と、ソフトメーカー、システムインテグレータ、コンサルタントとの間の取引慣行・契約に関する調査を行い課題を明確化する。

また、パッケージソフトウェア、ASP の企画、設計、構築、運用、保守にかかわる関係者（ユーザ、システムインテグレータ、メーカー、コンサルタント等）の権利と義務、責任の明確化及び信頼性向上に資するガイドライン、モデル契約の策定、政策、普及策の提言を行う。

１．３ ガイドの全体像

（１）位置づけ

本ガイドは、平成１９年４月に経済産業省が発表した「情報システム・モデル取引・契約書」を細くすることを目的に策定されている。パッケージや ASP のシステムライフサイクルの中の企画、設計・開発の工程を対象としている。



保守、運用及びセキュリティ・可用性については、別冊を参照されたい・

（２）対象とする内容

本ガイドでは、パッケージを主体としたプロセスを対象としている。このプロセスは SLCP-2007 に準拠する予定である。特に以下の３種類の契約を想定している。

- 企画支援業務、RFP 作成等の企画コンサルティング契約
- 要件定義、カスタマイズ、アドオン、テスト等の設計・開発契約
- 上記、企画コンサルティング契約と設計・開発の一貫契約

また、工程ごとに金額や仕様の確定を行うフェージング契約も対象とする。

特に、ベンダの取引慣行とユーザの予算に起因する齟齬などからトラブルが多数発生している、提案書、見積書等での確認事項などを重点に以下の項目を検討する。

- 仕様、性能、信頼性に関する説明事項、手順
- 保証内容、保証期間（プロダクツライフサイクル)、トラブル発生時の対応窓口、保証対象とならない場合の説明事項、手順
- 最終的に双方が確認する重要事項説明書等の合意プロセスの組み込み

ここで、パッケージを主体といっても様々な導入形態があることから、以下のパターンを検討対象とする。

- パッケージベース開発１（１アプリケーションによる開発）
  - ✓ 概要

    システム導入を１つのパッケージ製品により導入する形態
  - ✓ 例

ＥＲＰによる全社システム
人事システムのための人事パッケージ

- パッケージベース開発２（パッケージインテグレーション）
  - ✓ 概要

    複数の機能別パッケージを組み合わせて総合的な機能の実現を図るパッケージインテグレーション
  - ✓ 例

    販売管理パッケージと会計パッケージの連携
- ミドルウェア・インフラパッケージの使用
  - ✓ 概要

    ユーザが利用するアプリケーションにはパッケージを使わないが、システムの中核機能としてパッケージを使うシステム
  - ✓ 例

    データベース上でのシステム構築
    入出力制御パッケージの利用
- ＡＳＰ等サービス利用
  - ✓ 概要

    システムを自社で持たず、インターネット経由でサービス提供を受ける。
  - ✓ 例

    会計ネットワークサービス
    ＣＲＭネットワークサービス

（３）前提条件

本ガイドで記述する内容は、以下の条件を前提としている。

契約当事者

専門知識を有しないユーザ

専門知識を有する IT ベンダ、システムインテグレータ、コンサルタント

例　委託者（ユーザ）： 民間中小・中堅企業、市町村地方自治体

受託者（ベンダ）： IT ベンダ、システムインテグレータ、コンサルタント

開発モデル

経済産業省「情報システム・モデル取引・契約書」パッケージ選定モデルに準じる。また、パッケージ(ASP) のカスタマイズ等はウォータフォール型、アジャイル型、反復繰り返し型を想定する

対象システム

企業基幹系、情報系システム

プロセス

共通フレーム 2007 による標準化されたシステム企画・開発・運用・保守プロセスによる

マルチベンダ契約

工程分割発注に対応

留意事項

ユーザの IT リテラシーが高いとは限らないため、契約当事者間の情報の質と量、理解に対する十分な配慮が必要である。また、それを前提として契約合意に至るプロセスにも配慮を行う必要がある。

１．４ 用語

SLCP2007＋α（別途検討）

２．パッケージベース開発

２．１ 概要

パッケージベース開発（１アプリケーションによる開発）

必要とする情報システムを 1 つのパッケージにより構築する方式である。全社パッケージを ERP で導入する場合や、人事システムを人事機能専門の総合パッケージで開発する場合がこの方式である。

（１）メリット

１つのパッケージをベースにシステム開発をするため、システム開発が容易である。また、システム全体での検証がベンダ内で終わっているため信頼性が高い。

（２）デメリット

機能が不十分なことがある。また、全体のフレームは既に構築されているため、カスタマイズを行うにも制約がある場合がある。

パッケージベース開発（パッケージインテグレーション）

必要とする情報システムを複数のパッケージにより構築する方式である。販売や会計など業務分野ごとに最適なパッケージを選定し、パッケージの組み合わせでシステム開発する場合がこの方式である。

（１）メリット

業務ごとに最適のパッケージを選定するので、業務の専門性を高めたり、高度なサービスの実現が可能になる。また、不要な機能をセットで導入する必要が無いことから、システムをコンパクトに構成できる。

（２）デメリット

パッケージ間のデータの受け渡しなどに問題が生じることがある。また、ユーザインタフェースがそろわない場合も多い。

ミドルウェア･インフラパッケージの使用

システムを構成する要素として、パッケージを導入することがある。ユーザの目には触れにくいが、重要な、データ管理、出力などの機能を担うことも多い。

（１）メリット

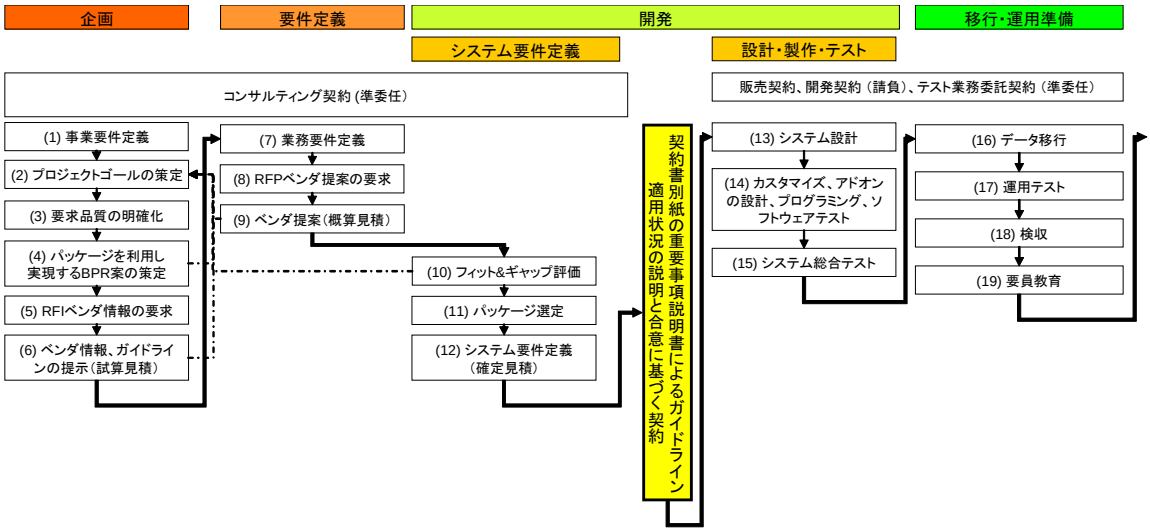
専門性の高い部位に安定した高度な機能を求めることができる。

（２）デメリット

システム内に組み込まれているため、障害の解析などが難しい。

２．２ モデル契約プロセス

プロジェクトのゴールを達成するために企画・設計を行っていくが、パッケージの持つ設計思想や機能により、ゴールそのものが変更になることもある。必要な機能をパッケージで実現できるのかどうかの検証を行うフィットギャップ分析が重要である。

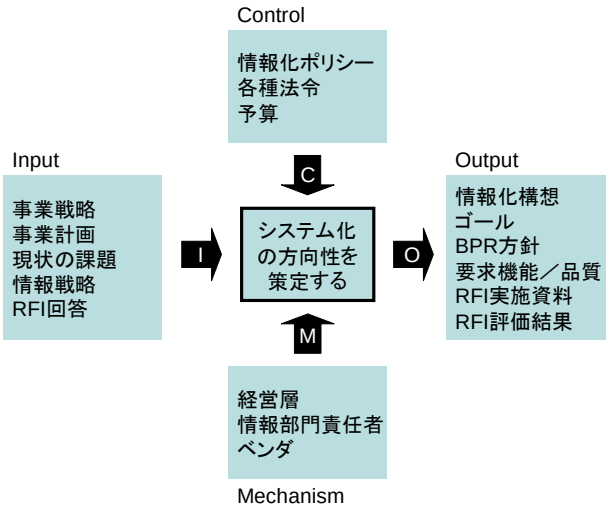


２．２．１ 企画プロセス

a. 企画（システム化の方向性）

パッケージの持つ知見やアイデアを調査し、現状の業務体系を見直す。プロジェクトゴール（システム導入の成果）を検討、策定するフェーズである。

①～⑥におけるプロセスは仮説設定、検証を目的としており、それぞれが独立したプロセスではなく、同時並行的、一体的に推移する場合もあり、大幅な手戻りや見直しが許容される。この時点からコンサルタントやベンダの参画を得る場合もある。

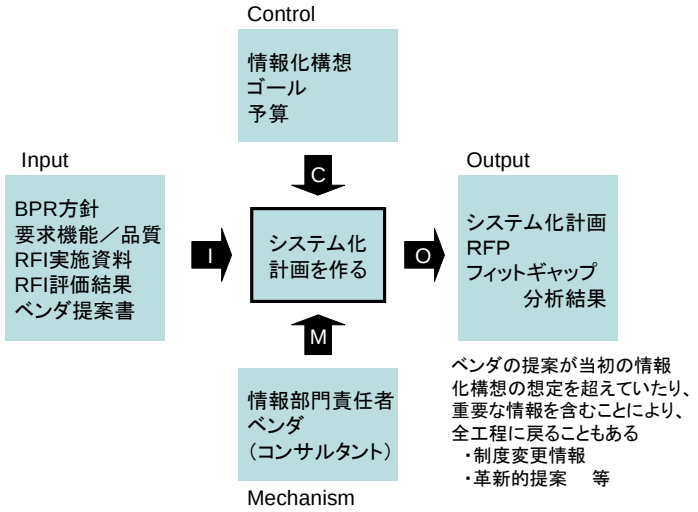


b．企画（システム化計画）

ＲＦＩによるベンダからの情報提供を踏まえブラッシュアップされた詳細なプロジェクトゴールや要求機能、ＢＰＲ方針をもとに、システムの全体方向性、ＢＰＲが検討されるフェーズである。

システム化計画が立案され、それを実現するための要求仕様書であるＲＦＰを作成する。ベンダからの提案を基にしたフィット&ギャップ評価では、パッケージの適合性、カスタマイズやアドオンの必要性、既存システムからの移行、要員教育、将来にわたる仕様の拡張性などとともに、運用・保守体制、償却期間におけるトータルコストと得られる効果を、パッケージごとに検証する。この工程は専門性も高いためコンサルタントやベンダの参画を得る場合も多い。

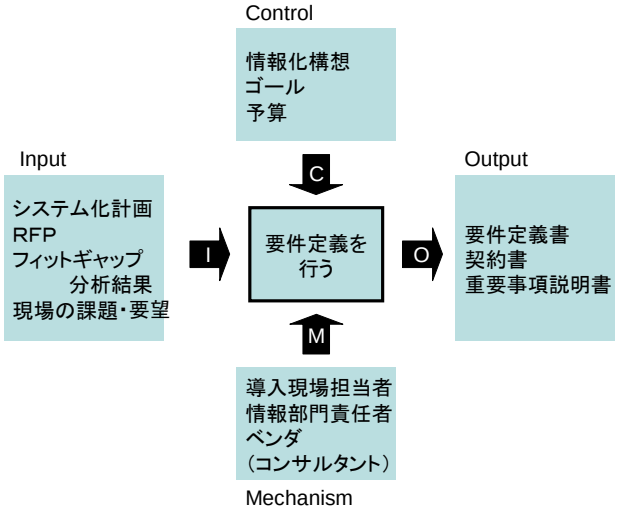
ここでも、フィット＆ギャップ評価に基づき、プロジェクトゴールやＢＰＲの見直しなど、上流工程にさかのぼった変更が許容される。



２．２．２ 要件定義

フィット＆ギャップ評価を得て、パッケージを選定した後に、必要なアドオン、カスタマイズ、システム構成等を整理し要件定義として取りまとめる。機能の詳細レベルでフィット＆ギャップ評価を行うこともある。

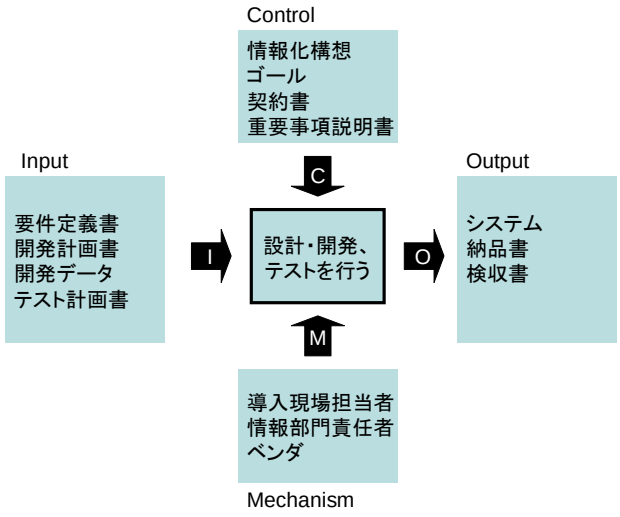
パッケージソフトウェアは一定の開発思想と経済合理性をもって設計されているため、ユーザ仕様に適合しない部分を無理にカスタマイズすることで、一部性能の大幅な低下や、将来の拡張に制限を招く場合があるため、カスタマイズによる制限事項や影響評価に関して留意する。



２．２．３ 開発

要件定義に従って設計を行い、実際にシステムを構築していくフェーズである。

開発プロセスは、パッケージベンダによるもの、パッケージそのものの開発者ではないが導入のみ行うシステムインテグレータによるもの、それぞれの再委託によるものと多岐に渡ってさまざまなケースが想定されるため、守秘義務と品質保証、受入テスト、検収に至る一連のフェーズについて、役割の明確化と十分な事前合意が必要である。


Control
情報化構想
ゴール
契約書
重要事項説明書
C
Input
要件定義書
開発計画書
開発データ
テスト計画書
I
設計・開発、
テストを行う
O
Output
システム
納品書
検収書
M
導入現場担当者
情報部門責任者
ベンダ
Mechanism

２．３ 企画・要件定義・開発で起こるトラブル例と必要な対応

企画設計に起こるトラブルは、ＲＦＰに起因する内容とその後の要件仕様確定によって起こることが多い。以下が多くの場合で指摘される主な問題点である。

- 不十分なＲＦＰ（要求仕様書）
- 過大なカスタマイズ、アドオン
- パッケージの機能・サービスレベル不足
- 仕様外の要求
- 検収時点での要求不一致
- 既存、追加ソフトウェアとの不整合
- 想定外の優越的地位の利用
- 知的財産の帰属
- 開発中止時の精算

（１）不十分なＲＦＰ（要求仕様書）

ユーザの作成したＲＦＰの内容が不十分であり、提案内容が絞りきれない場合がある。そのため提案内容に過不足が発生し、その後の要件定義の段階で詳細化しないまま進めると、設計途上や検収において、どちらの責任かもめることがある。特にユーザ側の意見として「プロならばＲＦＰや仕様の不十分さを補完せよ」、ベンダ側の意見として「仕様の確認は書面で交わしているので記述されていないことは対象外」といった議論が起こることが多い。

ユーザ側は、業務の暗黙知を仕様で表現できていないことも多く、また、ベンダの技術的な提案を理解できていない場合が多い。ベンダも、説明や聞きだす努力が不足していることもある。十分な配慮をする必要がある。

また、ＲＦＰの作成にコンサルタントを活用する場合もあるが、コンサルタントが十分な仕様を作成していない場合もある。コンサルティングを依頼するときには成果物イメージと記述レベルを十分に打ち合わせする必要がある。

	ユーザ	ベンダ	コンサルタント
原因	ＲＦＰの記述方法を知らない。 ＲＦＰを作る時間が取れない	不十分なＲＦＰを容認している ＲＦＰの内容を勝手に解釈している	ＲＦＰ作成支援において、十分なＲＦＰになっていない場合がある

主な対策	ユーザの業界に精通する第三者によるＲＦＰ評価・監査みたいなものを入れても良いのではないか	要件定義にする段階で十分な詳細化を図る	ユーザの業界に精通する第三者によるＲＦＰ評価･監査みたいなものを入れても良いのではないか
契約での留意点	ＲＦＰに記載されていない追加変更事項の扱いを契約書に記載する	同左	ＲＦＰに明確な不備があった場合の責任条項を記載する。（明確な不備とは、添付ＲＦＰ雛形の項目が記載されていないことや、当初ユーザが書面で提示した基本機能が記載されていない等、第三者評価で明確に不備と定義された事項） 紛争になった項目についてコンサルタントに証明義務を負わせる

（２）過大なカスタマイズとアドオン

フィットギャップ分析後にパッケージ選定を行うが、この後の詳細要件を定義していく中で、各種パラメータの設定、パッケージ本体に改造を加えるカスタマイズ、パッケージに機能を加えるアドオンを実施することとなる。このときにユーザが過大な要求をすることにより、システム全体のオリジナルのパッケージの占める割合が低くなり、システムの信頼性、安全性、性能が低下するとともに想定外の費用が発生することがある。また、カスタマイズの影響でバージョンアップが受けられなくなるトラブルも散見する。パッケージ選定後であっても、大規模改修が必要とわかった場合には、ＢＰＲ方針の見直しやパッケージ選定のやり直しなど、思い切った対策が必要な場合も出てくる点に留意が必要である。

	ユーザ	ベンダ	コンサルタント
原因	パッケージ導入のポイントを理解していない。 現場が既存の機能などに固執する	パッケージが提示する業務の優位性を十分に説明していない カスタマイズ増加が売り上げ増加につながる場合、容易にカスタマイズを受けてしまう	カスタマイズの必要性、投資対効果が説明できていない
主な対	パッケージ導入時には、パッケージ側の業務パターンにあわせる	企画段階でシステム化方針としてカスタマイズを少なくする方策を	カスタマイズに対する実施可否の評価基準やチェックリストを提示

策	ことも重要であることを利用現場に啓発する。	決定する 設計着手前に、十分なユーザ教育を行う	する
契約での留意点	契約上の留意点はない。	システムのバージョンアップに制限が入る事を契約書に明記する。 パフォーマンス低下の可能性を示唆する。	契約上の留意点はない。

（３）パッケージの機能・サービスレベル不足

フィットギャップ分析時に、当該機能有りと判断された機能について、設計段階においてユーザのイメージと大幅に異なることが判明し、カスタマイズの要否、その費用の負担などが問題になる場合がある。非機能要件において問題が発生することが多い。

特に、ある会社で作ったシステムをそのままパッケージと称して販売する流用品では機能の整備が十分ではないケースが見受けられる。システム構築の企画段階からパッケージを視野に入れて開発しシステム完成後に、パッケージ化を図る商品も、初期ユーザの仕様への依存度合いに注意が必要である

	ユーザ	ベンダ	コンサルタント
原因	パッケージ選定時の検討が不足している	パッケージの機能や制約を正確に説明していない	パッケージの精査が十分でない
主な対策	パッケージ導入時には、パッケージ側の業務パターンにあわせることも重要であることを利用現場に啓発する。	重要事項説明を実施する。 流用パッケージの場合には特に注意をする 性能などの非機能用件は早めの確認を行う	パッケージ選択の評価基準やチェックリストを提示する
契約での留意点	仕様に記述した機能が、既存機能より低下するときや一般常識的に用意されるべき機能が不足している場合の対応を明記する。	パッケージとして提供されるべき基本機能の不足等、紛争対応を明記する	同左

（４）仕様外の要求

システム開発途上で仕様に書いていなかったことが要求されることがある。それは、ユーザが仕様に書き忘れていた項目のときや、事業環境が変化したことに起因することも多い。ベンダとユーザの力関係で押し切られてしまうことも多く、手順の明確化が必要である。

	ユーザ	ベンダ	コンサルタント
原因	仕様の書き方や位置づけが理解できていない。 仕様確定後の利用部門からの要望をコントロールできない。	仕様確定時に十分な情報提供を行っていない	パッケージの精査が十分でない
主な対策	仕様のチェックリストを利用する。仕様の意味付けを利用現場に啓発する。	企画段階でシステム化方針としてカスタマイズを少なくする方策を決定する 設計着手前に、十分なユーザ教育を行う	パッケージ選択の評価基準やチェックリストを提示する
契約での留意点	仕様変更の費用分担判断基準や判断プロセスを明確にする	仕様に明記されていない部分での意識のすれ違いに関する検収基準を明確化しておく	同左

（５）検収時点での要求不一致

システム検収時点で現場の利用者が来て、これでは使えないなどの指摘をすることも多い。企画、設計などを情報システム部門にまかせっきりなユーザにおいて多発するトラブルであり、フローのあり方や操作性などの指摘を受けることも多い。仕様では明確に記述されていないことも多く、トラブルに発生することが多い。

	ユーザ	ベンダ	コンサルタント
原因	システム部門がユーザ部門と十分な確認を取っていない。 完成イメージを勝手に思い込み、ベンダに伝えていない。	完成イメージを勝手に思い込み、ユーザに伝えていない	ー
主な	プロトタイプやウォークスルーによるレビュ	プロトタイプやウォークスルーによるレビュ	ー

対策	ーを行う。	ーを行う。	
契約での留意点	契約上の留意点はない。	明文化されている仕様を実現していない場合はベンダの責に帰すべき要求不一致であり、明文化されていない仕様についてはベンダに帰すべき責でない旨を記載する。	ー

（６）既存・追加ソフトウェアとの不整合

システムの開発環境では問題なく稼動していたが、実運用機で動かしてみると既存システムと新システムで使用しているドライバが不整合などで正常に稼動しないことがある。逆に、追加システムを入れたことにより納入済のシステムが動かなくなることもある。また、セキュリティ対策などによるＯＳなどの変更もシステム障害を引き起こす可能性がある。

	ユーザ	ベンダ	コンサルタント
原因	稼働環境をベンダに事前に連絡していない。 テスト環境を提供しない。	事前に環境確認を行っていない。	ー
主な対策	事前に稼働環境をベンダに開示する。	事前環境調査を実施する	ー
契約での留意点	事前環境の確認を仕様に記述しておく。	システム開発終了後の別システム追加によるシステム障害は免責事項として記述する。	ー

（７）想定外の優越的地位の利用

ユーザとベンダは、本来は対等なパートナー関係を築く必要があるが、発注側であるユーザが相対的に優越的地位になることが多い。そのため、価格や仕様変更のリスクに備えフェージング契約をしていたとしても、後工程での金額の交渉ができない場合も多い。そのため、初期の概算見積もりのまま、開発をせざるを得なくなりトラブルになることがある。

また、グループ経営をしている企業の情報システムを構築するときに、企画まではグループの中核会社が実施し、その後、グループ会社と個別契約をすることもあるが、責任主体が明確でなくトラブルに発展することがある。

	ユーザ	ベンダ	コンサルタント
原因	当初合意していた事項を変更する。	契約条項などに記載していても、強く主張していない	ー
主な対策	合意内容を変更しない。	トラブルになりそうな項目は、契約書や議事録で確認を行う。	ー
契約での留意点	次フェーズで合意に至らない場合の解約条項を入れておく。 また、関連契約が影響を及ぼす場合は明記しておく。	次フェーズで合意に至らない場合の解約条項を入れておく。 また、関連契約が影響を及ぼす場合は明記しておく。	ー

（８）知的財産の帰属

著作権は開発者に帰属することが基本であるが、ユーザの知識不足による著作権保持の主張や、複数社が関わっての仕様検討により、業務フローなどのビジネスモデルについてユーザが権利を主張する場合もある。ビジネスモデルについては、特異なモデルではなく汎用的モデルについての権利を主張される場合、ベンダにとっては応じかねることがある。

	ユーザ	ベンダ	コンサルタント
原因	必要以上に著作権を主張する	業務秘密をパッケージなどで流用することがある	業務秘密を他社へのコンサルティングなどで流用することがある
主な対策	著作権と秘密保持契約の関係を啓発する	モラル啓発を行う。	モラル啓発を行う。
契約での留意	モデル契約を使って、常識的な契約を行う。必要であれば秘密保持契約を締結する。	業務秘密がユーザ固有の物である場合、汎用的で有る場合、業務秘密を流用した場合の扱いを契約書に明記する。	業務秘密を流用した場合の扱いを契約書に明記する。

点			

（９）開発中止時の精算

ユーザとベンダは、本来は対等なパートナー関係を築く必要があるが、発注側であるユーザが相対的に優越的地位になることが多い。そのため、価格や仕様変更のリスクに備えフェージング契約をしていたとしても、後工程での金額の交渉ができない場合も多い。そのため、初期の概算見積もりのまま、開発をせざるを得なくなりトラブルになることがある。

	ユーザ	ベンダ	コンサルタント
原因	仕様決定の遅れや頻繁な仕様変更など、ベンダが対応できない状況になってしまう。	技術力の不足などでプロジェクトが遂行できない。	左記両原因
主な対策	プロジェクト管理を強化する。	ベンダ選定を強化する。 プロジェクト管理を強化する。	プロジェクト管理を強化する。
契約での留意点	開発中止時の解約条件を明記する。	同左	同左

２．４ 関係者の役割分担

基本的にシステム利用者が責任を負うが、情報システム部門は技術面から支援をする必要がある。また、トラブルの多発するＲＦＰや要件の確定においては、現場とベンダが行力して行い、情報システム部門は調整を行う必要がある。

項目	ユーザ(現場)	ユーザ(ｼｽﾃﾑ)	ベンダ	備考
プロジェクトゴールの策定	◎	○	－	
要求品質の明確化	◎	○		
ＢＰＲ方針の策定	◎	○		
システム化の方向性	◎	○		
ＲＦＩ	◎	○		
ベンダ情報評価	◎	○		
システム化計画	◎	○		
ＲＦＰ	◎	○		
ＲＦＰの内容確認	○	△ 調整役	○	
ベンダ提案受付	◎	○		
フィットギャップ分析	◎	○	△ 支援	ベンダは必要な情報提供を行う
パッケージ選定	◎	○		
要件定義	◎	○		
要件の詳細項目確認	○	△ 調整役	○	
カスタマイズ・アドオンの設計、開発	◎	○		

コンサルタントが作業に参加するときには、ユーザのシステム部門と同じ立場になる。

ユーザ企業にシステム部門がいない場合

ユーザ部門の現場が責任を持つが、実質的にはコンサルタントが作業を行い、ユーザ部門は確認のみ行う場合が多い。

項目	ユーザ (現場)	コンサル	ベンダ	備考
プロジェクトゴールの策定	◎（確認）	◎（実質）	ー	
要求品質の明確化	◎（確認）	◎（実質）		
ＢＰＲ方針の策定	◎（確認）	◎（実質）		
システム化の方向性	◎（確認）	◎（実質）		
ＲＦＩ	◎（確認）	◎（実質）		
ベンダ情報評価	◎（確認）	◎（実質）		
システム化計画	◎（確認）	◎（実質）		
ＲＦＰ	◎（確認）	◎（実質）		
ＲＦＰの内容確認	◎（確認）	◎（実質）	○	
ベンダ提案受付	◎（確認）	◎（実質）		
フィットギャップ分析	◎（確認）	◎（実質）	△ 支援	ベンダは必要な情報提供を行う
パッケージ選定	◎（確認）	◎（実質）		
要件定義	◎（確認）	◎（実質）		
要件の詳細項目確認	◎（確認）	◎（実質）	○	
カスタマイズ・アドオンの設計、開発	◎（確認）	◎（実質）		

２．５ ベンダの説明事項と手順

２．５．１ 企画・要件定義プロセス

a. システム化の方向性

このフェーズで契約する場合の成果物は、情報化構想やシステム化の方向性に関する報告書である。報告書には、今後システム化を進めていく上での方針や要求すべき機能や品質の概要が記述される。

この作業は、社内のスタッフにより行われる場合も多いが、外部の専門家を検討に加える場合がある。準委任契約が行われる場合が多い。

この工程で外部支援を受けたときに起こる契約上のトラブルは以下の事項である。

- システム方針が経営者の満足する内容になっていない。

この工程におけるベンダの説明事項と手順、確認ポイントを以下に整理する

説明事項（報告書に下記項目を記載）

・システム方針（情報化構想）

システム全体の基本方針と将来構想を記述したもの。

・ゴール

業務やシステムを通じて実現したいゴールを記述したもの。

・ＢＰＲ方針

業務改革を行う基本的な視点や方針を記述したもの。

・要求機能・品質

この肯定を通じて成果として得られるものの品質を記述したもの。

保証内容、保証期間（プロダクツライフサイクル）、トラブル発生時の対応窓口、保証対象とならない場合に関する説明事項

ここで決める内容はあくまでも概要であり、保証の対象外とする

手順

- 提案時の説明事項、手順

提案時には、ユーザの要望をヒアリングしたベンダが、提案書として、ユーザの行いたいことを整理し、資料により確認

を行う。
確認項目は以下の内容である。
・背景
・目的
・実施したいこと
・目標指標
・実施手順
・成果物イメージ
・予算とその範囲
・体制と役割分担
また、提案確定時までに業務責任者の確認を受ける。

- 工程終了時の説明事項、手順

  最終成果物である報告書を基に、提案で記述された内容について説明していく。
  途中の合意を持って、提案内容を変えた場合には、その変更内容も含めて確認を行う。
  確認後、業務責任者の承認を受けることで終了する。

- 最終的に双方が確認するための合意プロセス

  契約形態に依存するが、基本的には、業務部門が作成したものを情報システム部門がレビューし、最終的に、業務責任者が承認を与えることが必要である。
  図で記述

運用・保守との連携に関する留意事項

このフェーズで運用・保守の詳細までは記述しないが、運用・保守が問題なく行える体制について検討をしていく必要がある。

セキュリティ・可用性に関する留意事項

最低限維持すべきセキュリティ・可用性の要件を明確化する。「ウイルス対策は行われていること」「＊＊機能は２４時間以内に復旧すること」などのように具体的に記述する。詳細レベルまでは記述する必要はない。

その他留意事項

なし

確認のポイント

提案時の確認のポイントは以下のとおりである。

チェック項目	Yes	No	備考
提案に成果物と記述レベルは明記されているか			
実施時のリスクは定義されているか			
・・・・・			
成果物としてシステムの方向性、ＲＦＩ、プロジェクトゴール等は得られるが、あくまでも概要に留まることに留意する。			

また、工程終了時の確認のポイントは以下の通りである。

チェック項目	Yes	No	備考
提案書に記述されている内容は実施されているか			
情報化の方向は計画にブレイクダウンできるレベルまで検討されているか			
・・・・・			
パッケージ以外の可能性も検討されているか			

b．システム化計画

このフェーズで契約する場合の成果物は、システム化計画書である。ＲＦＰが実施され、ベンダの提案に対するフィットギャップ分析結果も報告される。システム計画書には、システム化方式、機能の概要が記述されるとともに、具体的な実現方法、スケジュールなどが記述される。

この作業は、社内のスタッフにより行われる場合も多いが、外部の専門家を検討に加える場合がある。準委任契約が行われる場合が多い。

この工程で外部支援を受けたときに起こる契約上のトラブルは以下の事項である。

- システム計画書に具体性が無く、システム設計に詳細化していくことができない。
- システム計画書が、技術や期間などの面から実現不可能であり、システム設計に進めない
- 実施したＲＦＰの記述内容が不十分であり、後工程でトラブルが発生する

この工程におけるベンダの説明事項と手順、確認ポイントを以下に整理する

説明事項（報告書に下記項目を記載もしくは別添）

・ＲＦＰ

システムの概要や要求機能、品質などを記述するもの。

・フィットギャップ分析結果

ベンダからの提案内容の要求への整合状況、パッケージ選定の考え方について説明を行うもの。

・システム計画書

システムの具体的な計画が記述されたもの。

保証内容、保証期間（プロダクツライフサイクル）、トラブル発生時の対応窓口、保証対象とならない場合に関する説明事項

（オプション1）

ＲＦＰの記述内容や記述レベルはトラブルの最大の原因となる。ＲＦＰに関する内容の保証は契約上難しいので、十分な確認を行うことが重要である

（オプション２）

ＲＦＰやフィットギャップ分析が十分な検討のもとおこなわれなかったという

瑕疵が後工程で見つかった場合には、当該作業を実施したコンサルタントの瑕疵も指摘していく必要がある。ユーザ事由により要件未確定であったものの瑕疵を問うことはできないが、常識的に必要な検討が行われていないことによる問題はコンサルタントに責任を負わせる必要がある。当該部分の再調査もしくは当該部分相当費用の返還が必要になる場合もある。

手順

- 提案時の説明事項、手順

  提案時には、ユーザの要望をヒアリングしたベンダが、提案書として、ユーザの行いたいことを整理し、資料により確認を行う。
  確認項目は以下の内容である。
  - ・背景
  - ・目的
  - ・実施したいこと
  - ・目標指標
  - ・実施手順
  - ・成果物イメージ
  - ・予算とその範囲
  - ・体制と役割分担

  また、提案確定時までに業務責任者の確認を受ける。

- ＲＦＰ実施時の説明事項、手順

  ＲＦＰの意味、責任範囲、ＲＦＰで記述される要件の詳細、ＲＦＰの配布対象について説明を行う。特に、契約時に正式仕様を決定するが、そのもとになるのがＲＦＰであり、ＲＦＰがユーザの責任において作成される公式文書であることの理解を得ることが重要である。
  ＲＦＰの記載内容をチェックリストを使いながら確認を行う。

- 工程終了時の説明事項、手順

  最終成果物である報告書を基に、提案で記述された内容について説明していく。
  途中の合意を持って、提案内容を変えた場合には、その変更内容も含めて確認を行う。
  主な説明項目は以下の通りである。
  - ・ＲＦＰの記載内容
  - ・フィットギャップ分析とその分析結果と判断根拠
  - ・システム計画により初期目的が達成する根拠
  - ・システム計画が実施可能な根拠

  確認後、業務責任者の承認を受けることで終了する。

- 最終的に双方が確認するための合意プロセス

  契約形態に依存するが、基本的には、外部ベンダもしくは業務部門が作成したものを情報システム部門がレビューし、最終的に、業務責任者が承認を与えることが必要である。

運用・保守との連携に関する留意事項

このフェーズでＲＦＰにおいて、運用時間や必要な運用体制など、運用・保守の要件を記述する。

セキュリティ・可用性に関する留意事項

このフェーズでＲＦＰにおいて、セキュリティ・可用性の要件を記述する。要件は見積もりに影響するので、2 重化などのシステム構成に影響する内容、バックアップに対する対応やアクセスログなどの機能を付加するなど、一般的な脳よりも多くの機能を要する場合には必ず記入する。

その他留意事項

- ✓ 最低限確保されるべき応答時間や処理時間などの操作性に代表される品質要件は BPR に密接に関連し、技術要件にも大きな影響を及ぼすため、早期に優先度と重要度を明確にすることが望ましい。また、セキュリティ、既存システムからの移行、運用・要員教育、保守等において特段に配慮すべき項目もあわせて抽出しておくことが望ましい。特に、⑦システム化計画以降は、現場で運用に携わるオペレータや、利害関係者とプロジェクトゴールの共有や、システム化計画の周知と同意を得ておくことが重要で、プロジェクトチームとＢＰＲに直面する現場の調整に留意すべきである。
- ✓ パッケージに対するカスタマイズについては、そもそもパッケージが想定していない運用を求めることによって、パッケージの構造そのものの変更などがありえるため、カスタマイズやアドオン機能の工数と実現性について詳しく評価を行うべきである。
- ✓ フィット&ギャップ評価においてベンダの参加を求める場合、当該作業の内容と責任の所在を決めることが重要である。また、複数ベンダからの提案書およびフィット&ギャップ評価を求める場合は、書式の統一、用語の定義等に配慮し、相互理解に十分な時間をかけることが信頼性確保につながることに留意すべきである。
- ✓ パッケージベンダとシステムインテグレーションベンダが異なる場合、システムインテグレーションベンダで解決できない不具合や仕様変更が想定される。ベンダ間の協力体制、パッケージベンダとユーザにおける保守契約も合わせて選定評価の基準とすることが望まれる。
- ✓ パッケージの保守期間は、前提となる OS やハードウェアの世代交代、保守打ち切りに影響される場合がある。パッケージの保守期間、ハードウェアの保守期間とともに OS の動向、保守打ち切りの際の移行、費用についても事前に調査、想定することが望まれる。

c．要件定義

このフェーズで契約する場合の成果物は、要件定義書、契約書、重要事項説明書である。この作業により、今回構築するシステムの機能や開発方法などが明確に規定される。

この作業は、社内のスタッフにより行われる場合も多いが、外部の専門家を検討に加える場合がある。準委任契約が行われる場合が多い。

この工程で外部支援を受けたときに起こる契約上のトラブルは以下の事項である。

- 要件定義に具体性が無く、システム設計に詳細化していくことができない。
- 要件定義が、技術や期間などの面から実現不可能であり、システム設計に進めない

この工程におけるベンダの説明事項と手順、確認ポイントを以下に整理する

説明事項（報告書に下記項目を記載もしくは別添）

・要件定義書

システムの要求機能、品質などを規定するもの。

・契約書

ユーザとベンダの契約内容が明記された文書である。

・重要事項説明書

ベンダがユーザに説明し、同意を取るべき重要事項が記述された文書であり、契約書の付帯文書である。

保証内容、保証期間（プロダクツライフサイクル）、トラブル発生時の対応窓口、保証対象とならない場合に関する説明事項

要件定義書の記述内容や記述レベルはトラブルの最大の原因となる。用件定義書に関する内容の保証は契約上難しいので、十分な確認を行うことが重要である

手順

- 提案時の説明事項、手順

  提案時には、ユーザの要望をヒアリングしたベンダが、提案書として、ユーザの行いたいことを整理し、資料により確認を行う。

確認項目は以下の内容である。

・背景
・目的
・実施したいこと
・目標指標
・実施手順
・成果物イメージ
・予算とその範囲
・体制と役割分担

また、提案確定時までに業務責任者の確認を受ける。

- 要件定義書、契約書、重要事項説明書確定時の説明事項、手順

  ＲＦＰの意味、責任範囲、ＲＦＰで記述される要件の詳細、ＲＦＰの配布対象について説明を行う。特に、契約時に正式仕様を決定するが、そのもとになるのがＲＦＰであり、ＲＦＰがユーザの責任において作成される公式文書であることの理解を得ることが重要である。
  ＲＦＰの記載内容をチェックリストを使いながら確認を行う。

- 工程終了時の説明事項、手順

  最終成果物である報告書を基に、提案で記述された内容について説明していく。
  途中の合意を持って、提案内容を変えた場合には、その変更内容も含めて確認を行う。
  主な説明項目は以下の通りである。

  ・ＲＦＰの記載内容
  ・フィットギャップ分析とその分析結果と判断根拠
  ・システム計画により初期目的が達成する根拠
  ・システム計画が実施可能な根拠

  確認後、業務責任者の承認を受けることで終了する。

- 最終的に双方が確認するための合意プロセス

  契約形態に依存するが、基本的には、外部ベンダもしくは業務部門が作成したものを情報システム部門がレビューし、最終的に、業務責任者が承認を与えることが必要である。

運用・保守との連携に関する留意事項

このフェーズで要件定義書において、運用・保守の要件を記述する。

セキュリティ・可用性に関する留意事項

このフェーズで要件定義書において、セキュリティ・可用性の要件を記述する。

その他留意事項

- ✓ 最終仕様の決定においては、画面の遷移や帳票の形式、運用手順等を、現場オペレータの参画と承認を得るとともに、導入に備えた教育計画が必須である。導入後の手直しや変更を最低限に留めることは、信頼性向上とコストに重大な影響を及ぼすため、十分に留意する。
- ✓ パッケージに対するカスタマイズによって、基本機能に制限が加わるなどの他の機能に及ぼす影響と、非機能要件に及ぼす影響について確認し、要件定義とする。
- ✓ ハードウェア、ネットワークの高性能化、低価格化に伴い、データ量の増大とデータの分散が顕著である。信頼性要件、セキュリティ要件の観点から、要件定義の最終評価を行うとともに、これらが付随的要件でないことに留意し信頼性を確保されたい。

２．５．２ 開発プロセス

このフェーズで契約する場合の成果物は、要件定義書、契約書、重要事項説明書である。この作業により、今回構築するシステムの機能や開発方法などが明確に規定される。

この作業は、社内のスタッフにより行われる場合も多いが、外部の専門家を検討に加える場合がある。準委任契約が行われる場合が多い。

この工程で外部支援を受けたときに起こる契約上のトラブルは以下の事項である。

- 要件定義に具体性が無く、システム設計に詳細化していくことができない。
- 要件定義が、技術や期間などの面から実現不可能であり、システム設計に進めない

この工程におけるベンダの説明事項と手順、確認ポイントを以下に整理する

説明事項（報告書に下記項目を記載もしくは別添）

・要件定義書

システムの要求機能、品質などを規定するもの。

・契約書

ユーザとベンダの契約内容が明記された文書である。。

・重要事項説明書

ベンダがユーザに説明し、同意を取るべき重要事項が記述された文書であり、契約書の付帯文書である。

保証内容、保証期間（プロダクツライフサイクル)、トラブル発生時の対応窓口、保証対象とならない場合に関する説明事項

要件定義書の記述内容や記述レベルはトラブルの最大の原因となる。用件定義書に関する内容の保証は契約上難しいので、十分な確認を行うことが重要である

手順

- 提案時の説明事項、手順

  提案時には、ユーザの要望をヒアリングしたベンダが、提案書として、ユーザの行いたいことを整理し、資料により確認を行う。

確認項目は以下の内容である。

・背景
・目的
・実施したいこと
・目標指標
・実施手順
・成果物イメージ
・予算とその範囲
・体制と役割分担

また、提案確定時までに業務責任者の確認を受ける。

- 要件定義書確定時の説明事項、手順

ＲＦＰの意味、責任範囲、ＲＦＰで記述される要件の詳細、ＲＦＰの配布対象について説明を行う。特に、契約時に正式仕様を決定するが、そのもとになるのがＲＦＰであり、ＲＦＰがユーザの責任において作成される公式文書であることの理解を得ることが重要である。
ＲＦＰの記載内容をチェックリストを使いながら確認を行う。

- 契約書確定時の説明事項、手順

ＲＦＰの意味、責任範囲、ＲＦＰで記述される要件の詳細、ＲＦＰの配布対象について説明を行う。特に、契約時に正式仕様を決定するが、そのもとになるのがＲＦＰであり、ＲＦＰがユーザの責任において作成される公式文書であることの理解を得ることが重要である。
ＲＦＰの記載内容をチェックリストを使いながら確認を行う。

- 重要事項説明書確定時の説明事項、手順

ＲＦＰの意味、責任範囲、ＲＦＰで記述される要件の詳細、ＲＦＰの配布対象について説明を行う。特に、契約時に正式仕様を決定するが、そのもとになるのがＲＦＰであり、ＲＦＰがユーザの責任において作成される公式文書であることの理解を得ることが重要である。
ＲＦＰの記載内容をチェックリストを使いながら確認を行う。

- 工程終了時の説明事項、手順

最終成果物である報告書を基に、提案で記述された内容について説明していく。
途中の合意を持って、提案内容を変えた場合には、その変更内容も含めて確認を行う。
主な説明項目は以下の通りである。

・ＲＦＰの記載内容
・フィットギャップ分析とその分析結果と判断根拠
・システム計画により初期目的が達成する根拠
・システム計画が実施可能な根拠

確認後、業務責任者の承認を受けることで終了する。

- 最終的に双方が確認するための合意プロセス

  契約形態に依存するが、基本的には、外部ベンダもしくは業務部門が作成したものを情報システム部門がレビューし、最終的に、業務責任者が承認を与えることが必要である。

運用・保守との連携に関する留意事項

このフェーズで要件定義書において、運用・保守の要件を記述する。

セキュリティ・可用性に関する留意事項

このフェーズで要件定義書において、セキュリティ・可用性の要件を記述する。

その他留意事項

- ✓ 対象となるパッケージのプログラムは、(1)パッケージの基本機能部分、 (2)画面、帳票などのカスタマイズを前提としている部分、(3)新たに開発されるアドオン部分に分類される。プログラム変更、改修が(1)に係る場合は、信頼性に多大な影響を及ぼすとともに、将来に渡る保守が得られない場合もある。パッケージベンダと開発ベンダ、ユーザとの十分な相互理解と承認を得た上で、プログラム変更、改修を実施されたい。
- ✓ 受入テスト、検収については、実際の運用を想定したシナリオテストをベンダとユーザにおいて事前検討するとともに、シナリオテストに使用するデータによってテストの信頼性が大きく異なることから、使用データについてはベンダと十分な協議が望まれる。

２．６ パッケージインテグレーション型での補足事項

問題点
- パッケージインターフェースに起因する問題の解決が難しい
- そもそも障害が切り分けられない場合がある
- 原因がわかるまでの初期障害対応に問題がある

２．７ ミドルウェア・インフラパッケージの利用での補足事項

問題点
- アプリケーションと密接に関係し、バージョンアップができないときがある
- サポートが切れてしまうことがある
- ＳＩがいなくなってしまうと、扱いに困ることがある

２．８ 必要なドキュメント

RFI

RFP

契約書

仕様書

重要事項説明書

検収書

３．サービス調達（ＡＳＰ、ＳａａＳ）

３．１ 概要

ユーザが事故保有するシステムにアプリケーションを導入し、自社もしくは委託先に運用を委託する従来の情報システム購入モデルに対して、ネットワークを通じてベンダが所有するシステムのうえでサービスを利用し、業務を行う情報サービス利用モデルである。従来からもＡＳＰといわれるネットワーク型のサービスは提供されていたが、ＳａａＳでは、複数企業に同じプラットフォームを使ったサービスを行うマルチテナント方式を使うとともに、Three-Tier と呼ばれる。ユーザの画面、処理のプロセス、データベースが分離され、ユーザは画面表示のブラウザしか持たないモデルが一般的である。

		UI 画面表示、イベント発生	Biz Logic 画面遷移、データ処理、DB アクセス	DB DB 管理
	Stand-Alone	Client	Client	Client
ＡＳＰ	Two-Tier	Client	Client	Server
Ｓａａ Ｓ	Three-Tier	Client	Server	Server

システムの導入ではなくサービスの利用になるので費用の構造もこれまでと変わってくる。サービス利用契約を行っている期間に定額料金を支払う定額料金型とユーザ数やデータ量によって費用を変動させる変動型の課金が一般的に導入されている。

定額料金型

必要とする機能を月額等の利用サービスという形で調達する方式。比較的汎用性が高い業務にて主に利用される。グループウェア・バックオフィス系のアプリケーションが多い。

（１）メリット

導入の手間が少ない。

導入コストが少ない。

（２）デメリット

データなど重要な部分を外部に預けるため、セキュリティ面で不安。

自社での制御ができない。（サーバメンテナンスなど）

カスタマイズできない。（不必要な機能も一緒についている）

既存システムとの連携が難しい。

従量課金型

必要とする機能を使う分だけ契約し調達する方式。代表的なものにはホスティングサービスやストレージレンタルサービスがある。

（１）メリット

導入の手間が少ない。
利用する分だけの契約・導入なので不必要な機能がない

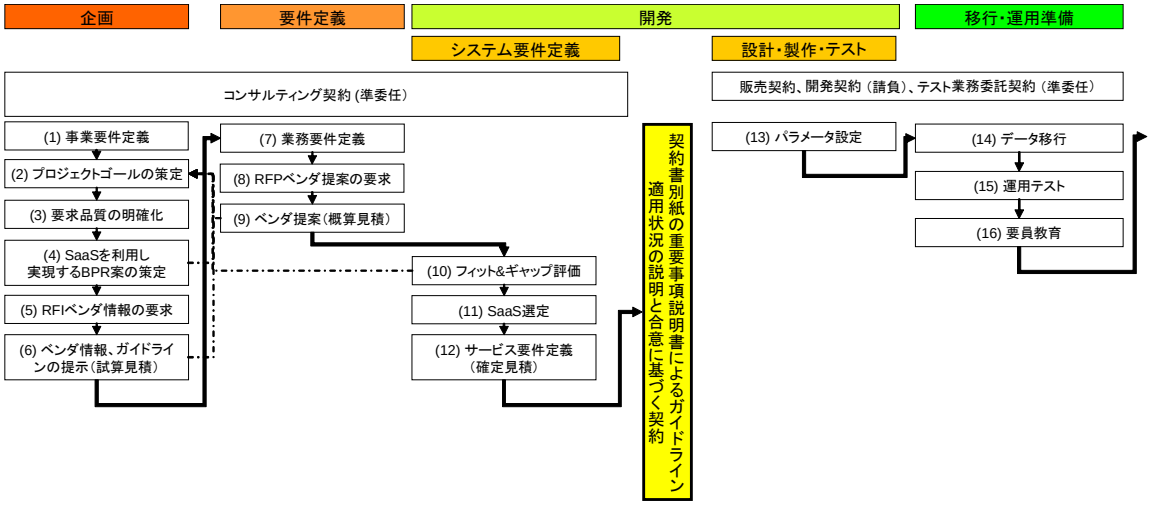
（2）デメリット
データなど重要な部分を外部に預けるため、セキュリティ面で不安。
自社での制御ができない。(サーバメンテナンスなど)
既存システムとの連携が難しい。

また、SaaS はネットワークを通じて情報システムによるサービスを受けるが、システム開発と同様の特徴がある部分と差異がある部分がある。契約という観点からその特徴を整理する。

	ＳａａＳ	システム開発	備考
データバックアップ	オプションサービスの有無による	自由に設定できる	
セキュリティ	ベンダの示した情報で判断	自分で設定	
サービス変更	月次等でできるところと、できないところがある	一旦開発したら減らすことはできない	
途中解約	可能なところと、一定期間は不可なところがある	一旦開発したら、使うしかない	
サービス停止時の対応	回復を待つしかない	自社での対応が可能	専門性を考えると自社で行うよりも他社で行ってもらったほうがよい場合もある
民事再生時の対応	サービス停止	事前調整可能	
天災時の対応	回復を待つしかない	自社での対応が可能	
ベンダの瑕疵	ベンダにより対応が分かれる	責任を問える	

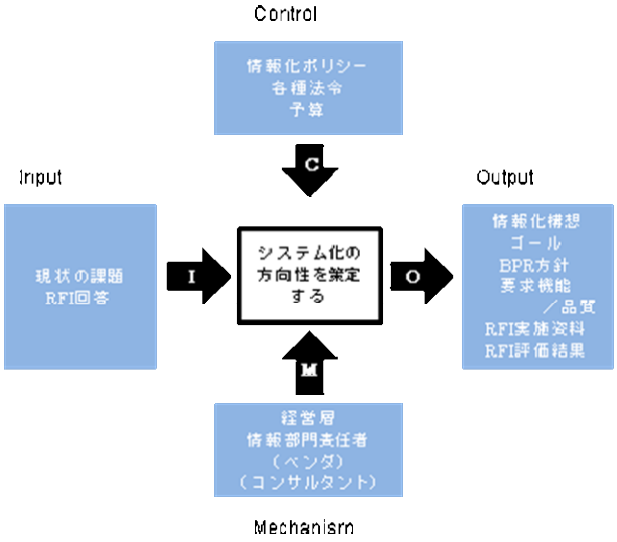
３．２ 主要プロセス

一般的なプロセスを記述



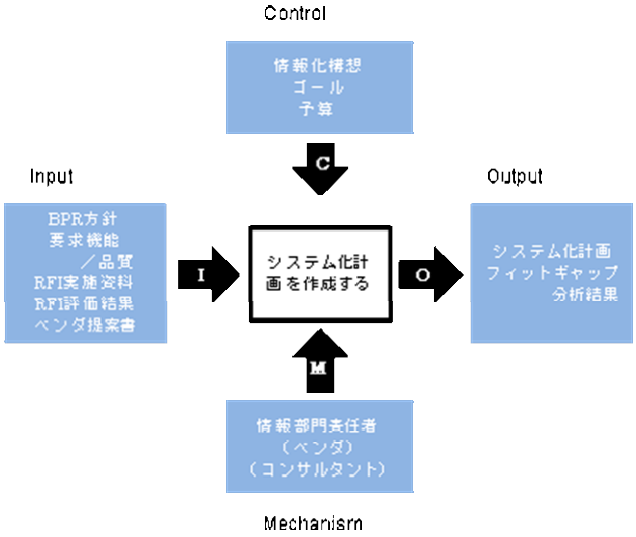
a.システム化の方向性

プロジェクトの目的を明らかにし、要求される機能を洗い出し、サービスの情報を収集し、システム導入の方向性を決定するフェーズである。ここでは、各ベンダのサービス内容を比較検討し、可能な機能・不可能な機能を明確にしておくことが重要である。また、場合によっては、プロジェクトゴールの策定をやり直す必要がでてくる。

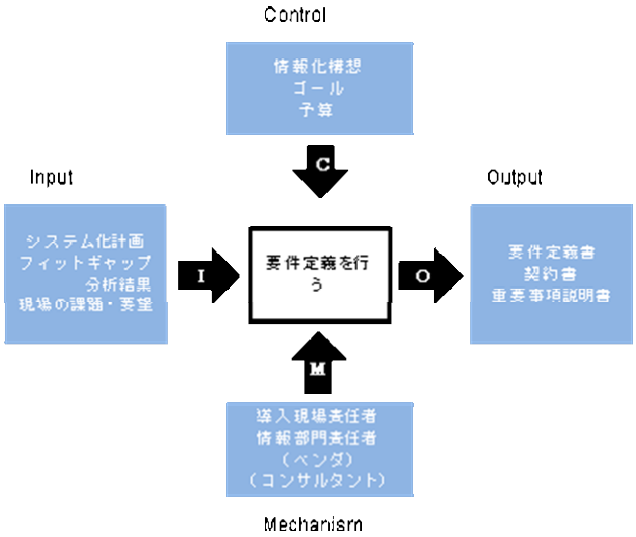


b.システム化計画

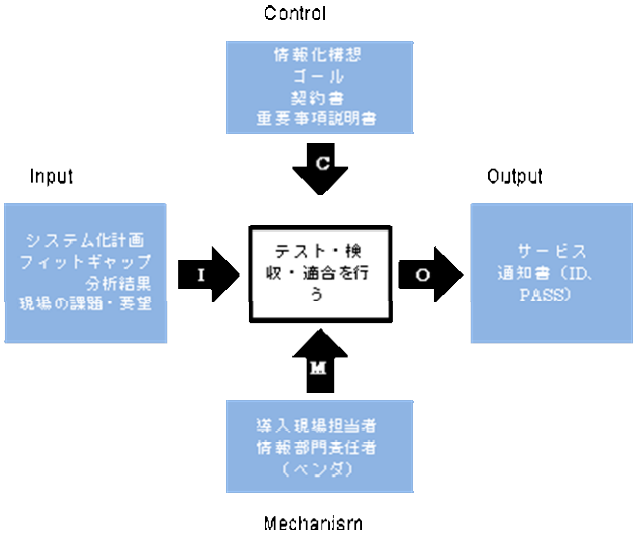
収集した情報をもとに具体的なシステム化計画を策定するフェーズである。ここでは、フィットギャップ分析を行い、要求品質に対する機能の過不足を明確にしておくことが必要である。必要に応じてプロジェクトゴールの策定やシステム化計画の策定をやり直す場合もある。



c.要件定義
システム化計画をもとに実際導入するサービスを選定するフェーズである。ここでは、サービスでカバーできない部分を明確にし、その対応策を考えておく必要がある。



d.検収プロセス
サービス契約を締結し、納品・検収作業を経て運用テストを行うフェーズである。導入前の最終段階となるので、実運用になるべく近い形でのテストが必要となる。



３．３ 企画・開発で想定されるトラブル等

これまでの ASP の導入、SaaS 導入を通じて、ユーザの不満や要求はあるものの契約上のトラブルはほとんど報告されていない。システムのようにこれから未知のものを構築するのではなく、既にあるサービスを納得して使うというのがＳａａＳのポイントであり、そのため契約上のトラブルまでは発展していない。しかし、利用者にまったく不満や不安がないわけではなく以下のような注意点に配慮する必要がある。

（１）機能不足

ユーザとベンダのコミュニケーション不足でサービスの機能をユーザが理解していない、もしくは誤解が生じる場合がある。ユーザ側の「こういう機能を期待していた」「標準的な機能かと思っていた」などの期待値がそのままクレームへと発展する場合があるため、ベンダは、サービスの機能について詳細にユーザ側に説明する必要がある。

（２）サービスの停止（ベンダ都合）

ベンダ側の都合（サーバメンテナンス、プログラム修正、バックアップ作業など）で一時的にサービスを停止する際に、「事前に聞いていなかった」「その時間帯はサービスをどうしても使いたい」といったユーザのクレームが起こる場合がある。ベンダは、契約時にその旨をしっかりと伝え、さらに停止の際にはユーザに事前に承諾もしくは通知する必要がある。

（３）サービスの停止（故障など）

災害時や通常環境でのハードウェアの故障、ソフトウェアの不具合などでサービスが停止してしまう場合がある。ベンダ側は停止しないよう最大限の努力を行う必

要があるが、発生してしまった場合は速やかに復旧の作業を行う。また、ベンダは復旧にかかる時間（保証時間）をサービス規約などであらかじめ明示しておく必要がある。

（４）サービスの中止

サービス提供会社の倒産などでサービスの維持が難しくなった場合、サービスそのものが中止される場合がある。事前に倒産時のサービス維持についての契約事項を盛り込むことが必要である。

（５）レスポンスの遅延

サービスレベル自体が通信環境に左右されるという特性を持っているため、ユーザ側の通信環境によっては満足いくレスポンス速度が得られないという場合がある。事前に実環境での運用テストを行う必要がある。また、ベンダは利用規約、場合によっては契約書にサービスの実行環境についての条件を明示する必要がある。

（６）個別環境（ハードウェア、ソフトウェア）での不具合

ユーザ側の個別の環境（ハードウェア、ソフトウェア）ではそれぞれソフトウェアやハードウェアの組み合わせによる競合によってサービスがうまく動かない場合がある。ユーザは、事前に実環境でのテストを行う必要がある。また、ベンダはあらかじめサービスの実行環境について詳細の条件を明示する必要がある。

（７）途中解約とスケールダウン

ユーザ側の都合により途中解約やユーザ数の減少などのスケールダウンをする場合がある。この場合の契約条件や手続きが明示されていないと、ユーザに予想外の契約金などが発生することがある

（８）データの保全

ベンダがデータの流出、誤消去などを行ったときの補償範囲を明確にしておかないと事故発生時に問題が発生することとなる

（９）データ移行

ユーザの都合でサービスの変更をするときにデータ移行のためのデータの変換または、移行ツールの提供がないと

（１０）マッシュアップサービスでの不整合

ユーザ側の都合により途中解約やユーザ数の減少などのスケールダウンをする場合がある。この場合の契約条件や手続きが明示されていないと、ユーザに予想外の契約金などが発生することがある

（１１）SaaS プラットフォームにおける障害

SaaS 提供企業が他社のプラットフォーム上でサービスを提供する場合、プラットフォーム停止時の責任をどのように問うのかが明確ではない。

（１２）価格改定

サービス利用途中での価格改定が行われると、ユーザにとっては計画外であるが、受諾せざるを得ない場合も多い。

３．４ 関係者の役割分担

システムの検討から導入まで一貫してユーザ責任によって作業を実施する。コンサルタントが導入を支援する場合でも、基本的にユーザ業務の実装だけが業務であり、また、利用開始後にユーザがサービス内容を変更することも可能なことからユーザの責任によって作業を行う。SaaS ベンダの責任範囲は利用規約に記述されている範囲内であり、その内容を十分に説明する義務のみを負っている。

３．５ ベンダの説明責任事項

（１）システム化の方向性
　　ベンダの参画がないため特になし。
（２）システム化の計画（導入計画）
　　サービス機能の詳細説明
　　利用時の規約（制限事項など）
　　安全性
　　データ保全方法
　　運用保証
（３）要件定義（オプションサービスの選択とパラメータの設定）
　　運用体制の説明
　　利用時の規約（制限事項など）
（４）検収プロセス
　　導入作業の流れ？

３．６ 必要なドキュメント

契約書

サービス利用規約（機能から利用時の環境まで）

重要事項説明書

４．開発手法別の留意点

この分野は、これまでのトラブル事例を見ていても契約では検討しきれない部分が多い。よって問題点の整理と概要説明のみとなる。

４．１ アジャイル開発・プロトタイピング

４．１．１ 概要

情報システムが企業内の業務で使われるだけではなく、インターネットを経由して一般の利用者にも直接利用されるようになって久しい。今では非常に多くの人がオンラインショッピング、オンラインゲーム、株などの金融取引や行政手続までをインターネットを経由して利用している。情報技術は日進月歩し、利用者の数は急激に増加した。新しいサービスも次から次に生み出されている。言い換えると日々進化する情報システム、新しい価値を提供できるシステム、予測できない事態に早急に対応できるようなシステムが求められるような情況になったといえる。また、オープンソフトウェアに代表されるように、実現する技術についても取捨選択していく時代になった。このような情況を考えると、従来のように結果をしっかりと予測し一から順番に積み上げていくウォーターフォール型の開発では予期せぬ事態に即座に対応できない、もしくはもっと別な方法のほうがより適切であるというようなケースも生まれてくる。そこで採用されることになったのがアジャイル型開発という考え方である。

アジャイル開発はまず目的と、その目的を果たすための主要な機能を実現し（プロトタイプ作成)、必要に応じて適宜修正を加えていく考え方である。常に利用者に対しては稼動する完成品の形を成している。当然、計画的に機能を追加していく（反復開発的要素）ようなケースもある。重要なことは目的につながるシステムの価値が常に提供され続けることである。アジャイル型開発における反復は、その結果として何らかの価値が提供されなければならない。

＜コメント＞アジャイル開発が向く「業務タイプ」について

完成したソフトウェアのメリデメについて

検討したが、特に業務タイプとして列挙することはあまり意味をなさないと考える。変化しないシステムであればＷＦであってもリスクはない。つまり「変化に対応するという目的があるのかどうか」が一つの答えになる。変化に対応しなくていいという業務を上げることは現在のビジネスシーンでは困難である。→　アジャイルがいいんだけどいろんな条件でやむなく［ＷＦ］というのが実態ではないかと考える。

同様に完成したソフトウェアに関してもアジャイルの方が目的を達成できないリスクが低いとは言いえるがそれ以上のものはないだろう。

４．１．２ 主要プロセス

アジャイル開発手法のプロセスを図示すると次のようになる。一旦完成したシステムは、さらに要求分析を繰り返し、改良を続けていくような形となる。ある意味では

完成することはないとも言える。

システムに求められる価値は常に提供され続ける。結果として時間をかければかけるほど価値のあるシステムとなることが求められる。一方で価値にならない機能を提供することがあってはならない。そのため、価値のある機能の検討精度を上げるためにも、リリース後の効果測定を行いながら要求を検討することも必要である。

＜注：出来上がるドキュメントについての記述＞

アジャイル開発手法を採用した場合にドキュメントが必要ないということは当たっていない。確かに出来上がったプログラムそのものがリーダブルであり成果物そのものということではあるが、取扱説明書のようなプログラムで表現できないドキュメントについては、それ以外のドキュメントが事前に作成されない関係上、より精緻なものが要求されることとなる。

４．１．３ 企画・開発時の要点

主要な参加メンバーが実現するサービスや業務について熟知していることが前提になる。さらに実装をするメンバーを含むメンバー間のコミュニケーションが重要になる。結果としてプロジェクトメンバーの数には制約が出る、参加できるメンバー個々の能力もそれなりに要求されることになる。

アジャイル開発手法を採用する場合には何を実現しようとしているのか、開発しようとしているシステムの目的はどこにあるのかというような点を十分に検討して採用すべきである。明確な目的が設定されていないと、要求を判断する際の指標が無いため、正しい判断が行えないためである。目的が明確でなければ、検討の方向性が曖昧になり、結果として出来上がるシステムも曖昧な形になることが多く、結果として無駄な投資がかさむことが歴史的にも証明されている。目的が複数設定されている場合は、目的に優先順位を設定することが求められる。実際のプロジェクトでは複数の目的が設定されていることも珍しくない。

目的に繋がる価値の検討において、しばしば議論になるのは、価値に繋がる要求の大きさである。ビジネスの価値とその要求の大きさは直交する関係にあるため、要求の大きさは大小まちまちである。また、この要求の大きさは開発計画に密接に関係する。

あらかじめ達成すべき目的が予測可能な時、つまり誰がいつ、どうような使い方を

して、どれくらいのトラフィックがあるのかなどがわかっているとき、また将来変動するような可能性が無いような場合にはアジャイル開発手法を採用するメリットは生まれてこない。

＜注：以下,XP を題材にした記述を追加＞

以下、アジャイル開発手法の代表的なものとされるエクストリームプログラミング（以下ＸＰ）について重要とされる点は以下のようなものである（参考：エクストリームプログラミング入門、wikipedia）

（XP における基本的事項）

ＸＰでは次の「４つを価値」が必要なものとして挙げられている。

１　コミュニケーション

- ✓　顧客、管理者、プログラマがそれぞれに対し即時に率直に質問できること。

２　シンプルであること

- ✓　動作させるために必要な最低限のところから始めて必要なものを付け加えている戦略をとること。

３　フィードバックが常になされること

- ✓　常にテストによって確認し、評価し、計画を修正し、結果を見せていくこと。

４　勇気を持つこと

- ✓　上記の３つを前提にして、大胆な取捨選択、改善、後退、挑戦を行う勇気を持つこと。

４．１．４ 関係者の役割分担

従来、ウォーターフォール型開発で可能であった開発依頼者が開発者に対して丸投げするような形は取ることが出来ない。開発依頼者はどういうものが出来上がるかということ、どういう機能を実現したいかということに対して常に決断と決断の責任を負う。開発者も要求される要件に対してその実現の可否の判断を速やかに行い、採用できる技術に基づき合意できる決着点を示す必要がある。したがってシステム化の対象となるビジネスを深く理解している人間が深く関与することはアジャイル開発には欠かせない。

同様に、アジャイル開発手法の代表的なものとされるエクストリームプログラミングについての関係者のプラクティスは以下のようなものである（参考：エクストリー

ムプログラミング入門、wikipedia）

１　チーム共通のプラクティス

- ✓ 通常１～２習慣の短い区切りの反復開発を行うこと。このことでリスクを最低限におさえ、常に稼動する状態を顧客に示していくことを可能にする。
- ✓ チーム全員の用語を共通化する
- ✓ 会話と作業の両方がしやすい環境を作る。
- ✓ フィードバックが行える環境、体制を作る。

２　開発のプラクティス

- ✓ 実装より先にテストを作成する（テストドリブン）。
- ✓ ペアプログラミングを行う
- ✓ リファクタリングを行う。
- ✓ ソースコードを共同所有する。
- ✓ 単体テストを通過するごとに結合テストを行う。
- ✓ 今必要なことだけを行う（ＹＡＮＧＩ）。

３　管理者のプラクティス

- ✓ 管理者の仕事を自分でちゃんとやる（責任の受容）。
- ✓ チームを教え、支援する
- ✓ 四半期ごとに見直しをする。
- ✓ チームの状態をメンバーに知らせる。
- ✓ 週４０時間労働を守る。

４　顧客のプラクティス

- ✓ ビジネスストーリー（カードー実現する機能等）を書く。
- ✓ リリース計画を提案しチームで合意し承認する。
- ✓ 受け入れテストをし、ストーリーが実現しているか確認する。実現していない場合は指摘する。
- ✓ 短期的にリリースする（リリース単位を小さくする）。

これらの中には単独でも機能するものを多く、部分的な採用も多く行われている。

但し、実装の部分に関連しているものが多く、要求定義の部分はビジネスストーリーとして与えられるという点には注意を要する。

４．１．５ ベンダの説明責任事項

開発依頼者に対して、採用する開発手法の選択に当たって十分な説明を行いその合意を得なければならない。さらに採用するにあたっては、開発依頼者とベンダー（開発者）側はお互いの役割分担とコミュニケーションの確立に当たってお互いに十分な理解を得なければならない。特に開発依頼者がアジャイル開発手法に対する理解、開発への参加の度合いが非常に濃密なものになることを理解することは不可欠である。

＜注：採用するときのチェックリストについては依頼中。一週間以内目途＞

４．１．６ 留意事項

反復繰り返し型 やアジャイル型開発と呼ばれる開発手法は、実際に機能するソフトウェアの提供に重点を置く手法である。実際に機能するという意味で、そのソフトウェアの実用度を早期に確認できるという特徴がある。

ウォータフォールモデルは上流から下流にむけて企画→要件定義→基本設計→詳細設計→運用テスト→運用というプロセスを踏むことから、万一、要件定義そのものに誤りがあった場合、その誤りはリリースまで判明しないという構造的欠陥が指摘されている。言い換えれば、ウォータフォールモデルでは、要件定義、外部設計でのレビューにおいて、設計書段階でのシミュレーションと確認を重ねることによって、手戻りを防止するモデルである。さらに、要件定義確定後の業務の変更や拡大が少なく、変更があっても予想可能な範囲にとどまるビジネスモデルであるとともに、発注者側が業務全般の説明能力に長けていることが信頼性確保の前提である。

反復繰り返し型（図 2）の場合は、上流から下流にむけての流れを繰り返すことによって、「統合・テストがすんで安定している『部分的に』完成したシステムをリリースすることである。」 とされている。言い換えれば、初期に仕様を完全に定義するのではなく、開発構想書や概要レベルの要求一覧表などをもとに、数週間単位で イテレーション（要求分析、設計、実装、テスト）を繰り返し、段階的に詳細化して仕様の完成度を高めていく開発手法である。イテレーションの流れはウォータフォールモデルに準じており、企画、要件定義の不備は運用テストで明らかになることから、これらの不備に対しては次のイテレーションの企画、要件定義で反映されることになる。同様にイテレーションにおける実装段階では、要件の変更は行わず、次のイテレーションでの課題として引き継がれていく。こうすることで、定められた期間での進捗と完成度を求め、全体の工数見積に反映させていく。

アジャイル型開発は要件定義そのものを、ユーザとともにプロトタイプの開発を通じて行ってしまおうという考え方にたつ。いわば、対話型のプログラムやユーザーインターフェースを実環境で実際に構築し、使い勝手やレスポンスの評価を得て、要件そのものを定義していくアプローチである。ユーザ自身で要件定義が困難であったり、ビジネスの範囲が成長、拡大している際に有効なアプローチである。反復繰り返し型異なるのは、オンサイト（ユーザのオフィスや実際のシステムの使用場所）開発、一

定期間の間に顧客が選択した機能を作りこむなどの、独特のプラクティスに従う点にある。

機能要件を決定し、非機能要件を満たすリソースを求めるという意味で、銀行のATM システムの開発においては、ウォータフォールモデルが明らかに優位である。あらかじめ、ATM で提供される機能は要件として定義が可能であり、画面のレスポンス等の非機能要件もユーザがビジネス上の重要な要件として定義することができる。こうした機能要件、非機能要件を実装できるハードウェア、通信環境、システム環境を選択し、必要なリソースを確保して開発にかかることで、運用テストにおける諸問題がどこにあるのかを追求するのも容易であると言える。

反面、ユーザ自身が機能要件を決定できない場合や、応答性や画面遷移といったユーザビリティが顧客満足度や生産性に影響を与えるシステム構築では、ウォータフォールモデルの紙上シミュレーションでは限界がある。反復繰り返し型開発やアジャイル型開発は、このような状況において適用が可能な開発モデルである。

アジャイル型（図 3）は、反復繰り返し型をもとに発展的にさまざまなプラクティスや原則を定義したもので、さまざまなモデルが存在している。モデルによっては大規模開発に向かないと言われているものもあるため、留意点が異なってくる。

・反復繰り返し型の場合でも、要件定義が信頼性に大きく影響を及ぼすことは同様である。開発構想書や概要レベルの要求一覧表で想定していなかった課題を要求事項として反映するための企画、要件定義プロセスをおろそかにすると、無用なイテレーションを繰り返すこととなる。また、次のイテレーションへのフィードバックを得るためのテストにおけるユーザの関与（評価）方法とテスト計画の変更管理、新たなイテレーションでの企画、要件定義の変更管理が必要である。一連の管理が不十分なままイテレーションが繰り返されると信頼性が大幅に損なわれることに留意する。

・要件定義の先送りなどによる無用なイテレーションを防ぐため、実現が必要な機能の優先順位の設定と、量的制限 を契約書に明記しておく必要がある。ユーザとベンダで、見積における積算根拠と作業結果の評価方法を事前に合意しておき、あらかじめ定めた反復回数で作業の見直しを行うことを契約に定めておくことが重要である。場合によっては、期間を定めた数回のイテレーションを見積のために契約し、その後、詳細見積、契約を締結するなどを検討すべきである。

・アジャイル型では、開発手法によってさまざまなプラクティスが提唱されており、ユーザとベンダがそのプラクティスを正しく理解することが信頼性確保においては重要である。代表的な手法であるエクストリーム・プログラミング(XP)では、オンサイト開発や、テスト手順を定めてから実装を行うテスト駆動型開発、反復ごとにユーザ

による受け入れテストを行うなどのプラクティスが定められており、従来のソフトウェア開発の慣行とは大きくかけ離れた特徴がある。開発者にとっても高い技術力と従来にない管理を要求することから、反復繰り返し型の留意点を踏まえ、ユーザ、ベンダの双方のプラクティスの実現方法、管理方法を契約書に明記する必要がある。

・動作するシステムから得られる効果を検証しながら要件定義が確定し、プログラムが完成した段階で設計が完成するため、ソースコードの保守性、可読性を高めるためのリファクタリングと呼ばれる作業が、信頼性向上においては大きく影響を及ぼす。リファクタリングの実現方法について、ユーザ、ベンダでの詳細な取り決めは、将来の保守性を高めるために有用である。

・反復繰り返し型、アジャイル型を含め、プラクティスにドキュメントが定められていないことを理由にドキュメントを作成しないことは誤りである。可読性の高いコードの創出とともにユーザにおける信頼性、保守性を確保するためのドキュメントの作成は重要である。XP においてもテクニカルライタの重要性が指摘されており 、プロジェクトの特徴と優先順位を鑑みたドキュメントのあり方を、ユーザ、ベンダで合意することに留意されたい。

４．２ 繰り返し型開発

４．２．１ 概要

システム開発を行うとき、さまざまな事情から、最初にすべて実現すべき内容が決定できるとは限らない。そのような場合、すでに内容が決定している主要な機能から順次、開発、実現していくような形が考えられる。実際のシステム開発は現状ではウォーターフォール型の開発といえども何らかの意味で繰り返し型開発的な要素を持っているといえる。大きな違いは、開発されるユニットを 1 つの価値としてそれごとに確立していくという点である（契約単位にすること）。

４．２．２ 主要プロセス

次に図示するようなプロセスである。

４．２．３ 企画・開発で起こるトラブル

どういうシステムが求められているのか明確に開発依頼者と開発者が合意しなけ

ればならないことなどウォーターフォール型開発と基本的にまったく同様である。利用者に対してリリースできるレベルでテストまでを含むプロセス単位を小分けにするので、プロジェクトとしてのリスクを軽減することが出来る。

４．２．４ 関係者の役割分担

ウォーターフォール型の開発と基本的に同様である。

４．２．５ ベンダの説明責任事項

ウォーターフォール型の開発と基本的に同様である。

５．モデル契約書・逐条解説

この章では、上位委員会で既に解説別の項目については参照のみを行い、新たに付け加えた事項のみ解説を行う。